RESCU: Dynamic Hybrid Packet-Loss Recovery for Video Transmission over the Internet[*]

Srinath R. Joshi, and Injong Rhee
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-7534
{srjoshi,rhee}@csc.ncsu.edu

Abstract

The current Internet is not reliable; packet loss rates are frequently high, and varying over time. Transmitting high-quality interactive video over the Internet is challenging because the quality of compressed video is very susceptible to packet losses. Loss of packets belonging to a video frame manifests itself not only in the reduced quality of that frame but also in the propagation of that distortion to successive frames. This error propagation problem is inherent in many motion-based video codecs due to the interdependence of encoded video frames. This paper presents a dynamic loss recovery scheme, called RESCU, to address the error propagation problem. In this new scheme, picture coding patterns are *dynamically* adapted to current network conditions in order to maximize the effectiveness of hybrid transport level recovery (employing both forward error correction and retransmission) in reducing error propagation. Since RESCU does not introduce any playout delay at the receiver, it is suitable for interactive video communication. An experimental study based on actual Internet transmission traces representing various network conditions shows that dynamic hybrid RESCU exhibits better error resilience and incurs much less bit overhead than existing error recovery techniques such as NEWPRED and Intra-H.261.

Introduction

Packet losses are fairly common in the Internet. During busy time, about 5% to 10% packet losses over the connections between the east and west coasts of US, or over the trans-atlantic or trans-pacific connections are not unusual. Packet losses occur mainly due to congestion (or lack of bandwidth capacity) on the path from the source to the destination. Since packet losses in the current best-effort Internet cannot be avoided, applications like Internet-based video telephony must make provision against packet loss to minimize its impact on their performance.

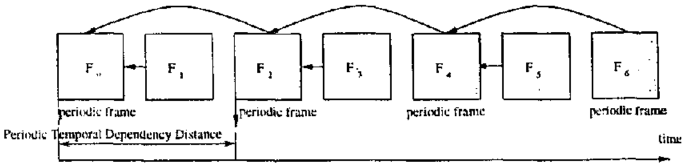

Figure 1: RESCU with PTDD 2 (Arrows represent temporal dependency)

The quality of compressed video is very susceptible to packet loss or corruption because of the way video signals are compressed. For instance, typical video encoders (MPEG-I, MPEG-II, H.261, H.263) use motion estimation and compensation in their compression algorithms. Motion estimation removes temporal redundancy in successive video frames (inter frame) by encoding only pixel value differences between the current image and its motion-predicted version reconstructed using a previously encoded image (reference frame or R-frame). Image distortion in a reference frame due to packet losses can propagate to its succeeding frames.

Recently a new scheme called *Recovery from Error Spread using Continuous Updates* (RESCU) is proposed[1], Unlike other existing conventional approaches, RESCU focuses on eliminating error propagation using transport level recovery rather than preventing errors from happening in the first place. In today's Internet, where packet losses and long network latency are common, recovering lost packets before the display times of their frames is not always possible. Therefore, some repair packets may arrive after the display times of their frames. While the conventional techniques discard these "late" repair packets, RESCU can use them to stop error propagation. This is facilitated by buffering displayed frames, and later restoring them whenever repair packets arrive.

The main benefit of RESCU is that it allows more time for transport-level recovery to succeed; repair packets for a frame are useful until that frame is being used as a reference frame. In order to accommodate recovery delays, RESCU designates every p-th frame as a *periodic frame*. The number of frame intervals between two consecutive periodic frames is called *periodic temporal dependency distance* (PTDD). Non-periodic frames within a PTDD period reference only their immediately preceding periodic frame. Figure 1 shows an example of RESCU picture coding patterns with PTDD 2. When a periodic frame undergoes packet losses, if its repair packets (retransmitted or FEC packets) arrive before the decoding of the next periodic frame, error propagation can be stopped. Also, since no frames reference non-periodic frames, loss in non-periodic frames does not cause any error propagation. No attempts need be made to restore non-periodic frames after their display and therefore, only periodic frames being recovered need to be buffered.

---

[*] This work is supported in part by NSF CAREER, ANIR-9875651.

SIGMETRICS 2000 6/00 Santa Clara, California, USA

Note that PTDD does not affect frame playout times because all frames are still displayed at their scheduled display times.

Problem Statement

Network conditions vary over time; congestion, transmission latency, loss rates, and available bandwidth frequently change. As network conditions change, the effectiveness of transport-level recovery (retransmission and FEC), and thus the associated recovery delays change. We need to adjust PTDD depending on the latest conditions to maximize the error resilience of RESCU. Note that PTDD cannot be set arbitrarily large because it reduces compression efficiency. Thus, finding the minimum PTDD under given network conditions that maximizes the periodic frame recovery probability is an important component of RESCU. This problem had not been addressed in our prior work, and is the subject of this paper.

Packet loss rate, loss burst length, and transmission delays play an intricate role in determining PTDD. For instance, if the loss rate increases, then either more FEC packets or retransmission attempts are required to maintain high loss recovery probabilities. Thus PTDD has to be extended to accommodate these increased number of repair attempts. Furthermore, loss burst characteristics significantly affect PTDD. As network traffic undergoes more bursty losses, retransmission becomes more effective than FEC since repair packets are transmitted only when losses occur. Hence, the loss burst characteristics influence the decision to use FEC or retransmission or both for recovery. When retransmission is used, PTDD must be at least as long as one round trip time (RTT). When FEC is used, PTDD should be at least as long as the product of the time interval between two consecutive FEC packets and the number of FEC packets required for protecting a periodic frame. When a hybrid technique combining FEC and retransmission is used, finding minimum PTDD becomes even more complex.

Contribution

For hybrid recovery (employing both retransmission and FEC) we give an algorithm to find minimum PTDD that is sufficiently large to achieve the desired recovery probability of periodic frames based on predicted network conditions. The network conditions during the next PTDD period are predicted based on the knowledge of the past network behaviors. This algorithm relies on exhaustive search around its parametric space to find minimum PTDD (we call it *exhaustive hybrid RESCU*). For experimental evaluation we also create two variants of this hybrid scheme, each using either FEC or retransmission (but not both) as a transport level recovery mechanism. The advantage of predictive modeling of future events as used in this hybrid scheme is that when network conditions are stable and predictable, it can achieve very high error resilience with controlled bit overhead. This scheme requires only one frame buffer in the decoder. However, when network conditions are highly bursty and unpredictable, PTDD estimation based on predictive modeling of future events tends to waste bandwidth because bit overhead is incurred even if (predicted) packet losses do not occur. In addition, since this algorithm requires the calculation of recovery probability, it may be computationally expensive. Thus, exhaustive hybrid is more suitable for an QoS controlled environment such as DiffServ where network characteristics are more predictable and not varying frequently.

To overcome some of the disadvantages of the exhaustive algorithm, we develop another hybrid recovery algorithm, called *lazy hybrid RESCU*. This algorithm adjusts PTDD for retransmission when the actual packet losses cannot be recovered through a proactive recovery technique such as FEC alone, so as to minimize the chances of ensuing frames being affected by the losses in the periodic frame. The main advantage of this strategy is that even if a proactive technique fails, it can still recover from losses. This allows lazy hybrid RESCU to budget only a small amount of bit overhead for proactive recovery. Since further bit overhead due to retransmission is incurred only when unanticipated bursty losses occur, this algorithm can outperform the exhaustive hybrid algorithm when the packet loss characteristics exhibit a high degree of variability, such as in today's Internet. This scheme is very computationally efficient and thus, suitable as an on-line protocol. However, this recovery scheme needs more frame buffers at the encoder and the decoder than the exhaustive recovery scheme which needs just one additional frame buffer at the decoder.

We have implemented each of the developed algorithms into H.261, and validated their effectiveness through simulation experiments. These experiments are designed using actual Internet traces obtained from video transmission experiments conducted over a transpacific connection at every hour over a two week period. The effectiveness of an algorithm is evaluated based on the end video quality measured by the peak signal-to-noise ratio (PSNR) and the bit rate measured by the number of bits transmitted.

We compare the performance of the dynamic RESCU schemes (exhaustive hybrid RESCU, its FEC and retransmission variants, lazy hybrid RESCU) with other error recovery techniques such as NEWPRED[2], and Intra-H.261 [3]. We use three video sequences originally used in MPEG-4 class A, B and E tests. The summary of the experimental results is given as follows:

Exhaustive v.s. Lazy. For every class of video sequences and the tested loss rates, the lazy hybrid RESCU performs as well as or better than the exhaustive hybrid RESCU. Their end video quality is typically less than 0.5dB different while the lazy has up to 10% less bit overhead than the exhaustive.

Hybrid v.s. FEC or retransmission. The hybrid RESCUs can take advantage of the respective strengths of FEC-only RESCU and retransmission-only RESCU; with up to 20% less bit overhead, the hybrid give only slightly worse PSNR (typically within 0.5dB) than the FEC-only and much higher PSNR than retransmission-only.

Hybrid v.s. NEWPRED or Intra-H.261. When compared to NEWPRED and Intra-H.261, under low to medium motion video sequences (classes A and B), the hybrid give up to 3dB to 4dB higher PSNR with 20% to 40% less bit overhead. Under a high motion video sequence (class E), the performance advantage of the hybrid is much attenuated. However the performance of the hybrid RESCUs is still better than the other techniques.

REFERENCES

[1] I. Rhee. Error control techniques for interactive low-bit rate video transmission over the internet. In *Proceedings of the ACM SIGCOMM'98*, pages 290–301, Vancouver, Canada, Sept. 1998.

[2] LBC. Document, LBC-96-033 (ITU-T SG 15, WP 15/1), An error-resilience method based on back channel signalling and FEC, 1996.

[3] S. McCanne and V. Jacobson. vic: a flexible framework for packet video. In *Proceedings of ACM Multimedia'95, San Francisco, CA*, pages 511–522, November 1995.